H

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/825,434	04/03/2001	Charles David Callahan II	324758006US2	2691

25096 7590 06/07/2004

PERKINS COIE LLP
PATENT-SEA
P.O. BOX 1247
SEATTLE, WA 98111-1247

EXAMINER	
RAMPURIA, SATISH	
ART UNIT	**PAPER NUMBER**
2124	

DATE MAILED: 06/07/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)
	09/825,434	CALLAHAN ET AL.
	Examiner	Art Unit
	Satish S. Rampuria	2124

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *03* MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *03 April 2001*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-129* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1,3-40,43,44,46-48,56-71,74-83,85-95 and 101-129* is/are rejected.

7)☒ Claim(s) *2,41,42,45,49-55,72,73,84 and 96-100* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *05/29/2002*.

4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application (PTO-152)

6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to the application filed on 04/03/2001.

2. Claims 1-129 are pending.

Information Disclosure Statement

3. An initialed and dated copy of Applicant's IDS form 1449 is attached to the instant Office action.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claim 1-32, 41-42, 49-62,73-79, 83-84, 88-90, 92, and 94-129 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent No. 6,055,492 to Alexander, III et al., hereinafter called Alexander, in view of admitted prior art, further in view of US Patent No. 5,903,730 to Asai et al., hereinafter called Asai, and further in view of US Patent No. 5,333,280 to Ishikawa et al., hereinafter called Ishikawa.

Per claim 1, 6-12, 18-21, 31-32, 62, 74, 79, 83, 90, 92, 94, 103, 104-109, 115-117, and 127:

Alexander disclose:

- A computer-implemented method for analyzing trace information generated during execution of multiple threads of a software program on a first computer (col. 2, lines 14-

17 "representing program event trace information in a way which is very compact and efficient, and yet supports a wide variety of queries regarding system performance" and col. 2, line 28 "stacks observed during a program's execution")

- receiving an indication of trace information reflecting a series of events that occurred during the execution (col. 3,line 66 "using trace file as input" and col. 2, line 28 "during a program's execution"), and each event having associated values in the trace information of variables maintained by the executing software program, by the one protection domain, and/or by the one processor (col. 1, lines 42-45 "a trace tool may log the requester and amount of memory allocated for each memory allocation request. Typically, a time stamped record is produced for each such event")
- for each of a plurality of periods of time during which the execution was occurring, determining from the trace information a number of instructions executed for the software program during the period of time by (col. 1, lines 42-45 "a trace tool may log the requester and amount of memory allocated for each memory allocation request. Typically, a time stamped record is produced for each such event")
- identifying multiple protection domains that each executed at least one of the multiple threads during at least a portion of the period of time; (col. 1, lines 57-59 "traces are many millions of entries long. Because loops are common in programs, trace files often contain patterns of events repeated many times")
- identifying processors that each executed at least one of the multiple threads during the period of time (col. 1, lines 57-59 "traces are many millions of entries long. Because

loops are common in programs, trace files often contain patterns of events repeated many times");

Alexander does not explicitly disclose the first computer having multiple processors that each have multiple protection domains that are each able to execute at least one of the multiple threads, each processor having a counter indicating a number of instruction holes during which an instruction is not executed by the processor, each protection domain having a counter indicating a number of instructions issued in the protection domain by all executing threads, each event associated with execution of one of the multiple threads by one of the protection domains of one of the processors, for each of the identified protection domains, determining a change in the value of the issued instructions counter of the protection domain during the period of time; determining if all of the instructions issued in the protection domain during the period of time were for one of the multiple threads; for each of the identified processors, determining a change in the value of the instruction holes counter of the processor during the period of time.

However, admitted prior art discloses in an analogous computer system the first computer having multiple processors that each have multiple protection domains that are each able to execute at least one of the multiple threads, each processor having a counter indicating a number of instruction holes during which an instruction is not executed by the processor, each protection domain having a counter indicating a number of instructions issued in the protection domain by all executing threads (Applicant's specification, prior art; page 1, lines 21-25 "Each processor contains a complete set of registers 101a for each stream such that the register values at any given time indicate the current stream state… each processor also supports multiple

protection domains, each with counters reflecting the current protection domain state 101b"), each event associated with execution of one of the multiple threads by one of the protection domains of one of the processors (Applicant's specification, prior art; page 3, line 14 "Each task (i.e., executing…) have one or more threads… executing… assigned to a protection domain in which task is executing") for each of the identified protection domains, determining a change in the value of the issued instructions counter of the protection domain during the period of time (Applicant's specification, prior art; page 3, lines 25-27 "The current numbers of streams executing in the protection domain is indicated by scur"); determining if all of the instructions issued in the protection domain during the period of time were for one of the multiple threads (Applicant's specification, prior art; page 3, lines 25-27 "The current numbers of streams executing in the protection domain is indicated by scur"); for each of the identified processors, determining a change in the value of the instruction holes counter of the processor during the period of time(Applicant's specification, prior art; page 3, lines 25-27 "The current numbers of streams executing in the protection domain is indicated by scur").

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the method of having counter for each protected domain, executing multithread task keep track of number of instruction to be executed and has been executed as taught in admitted prior art in corresponding to analyzing trace information as taught by Alexander. The modification would be obvious because of one of ordinary skill in the art would be motivated to have counter for each protected domain, executing multithread task keep track of number of instruction to be executed and has been executed to provide efficiency during the execution of the parallalism as suggested by admitted prior art (page 12, lines 1-2).

Neither Alexander nor admitted prior art explicitly disclose when it is determined that all of the instructions issued in the protection domain during the period of time were for one of the multiple threads, calculating a value for the number of instructions executed for the software program during the period of time by the protection domain to be the determined change, and when it is determined that all of the instructions issued in the protection domain during the period of time were not for one of the multiple threads, calculating a value for the number of instructions executed for the software program during the period of time by the protection domain to be a portion of the determined change that corresponds to a portion of the period of time during which at least one thread for the software program was executing in the protection domain; and determining the number of instructions executed for the software program during the period of time to be a sum of the calculated values for each of the identified protection domains; and if all of the instruction holes that occurred during the period of time were attributable to the software program, calculating a value for the number of instruction holes for the processor that are attributable to the software program during the period of time to be the determined change in the value of the instruction holes counter, calculating a value for the number of instruction holes that are attributable to the software program during the period of time by all of the identified processors to be a sum of the calculated values for each of the identified processors; and determining the number of instruction slots available for execution of the instructions of software program during the period of time to be a sum of the determined number of instructions executed for the software program during the period of time and of the calculated value for the number of instruction holes that are attributable to the software program

during the period of time; and presenting to a user an indication of the determined number of executed instructions for each of the periods of time and an indication of the determined number of available instruction slots for each of the periods of time.

However, Asai discloses in an analogous computer system when it is determined that all of the instructions issued in the protection domain during the period of time were for one of the multiple threads, calculating a value for the number of instructions executed for the software program during the period of time by the protection domain to be the determined change (col. 6, lines 49-51 "Calculating the average accumulated time of each parallel processing library routine called by each procedure according to the equations") and when it is determined that all of the instructions issued in the protection domain during the period of time were not for one of the multiple threads, calculating a value for the number of instructions executed for the software program during the period of time by the protection domain to be a portion of the determined change that corresponds to a portion of the period of time during which at least one thread for the software program was executing in the protection domain (col. 6, lines 49-51 "Calculating the average accumulated time of each parallel processing library routine called by each procedure according to the equations" and col. 6, lines 15-19 "Obtaining the final value of the loop control variable and calculating the number of iterations based on the initial and final values of the loop counter variable and the step size"); and determining the number of instructions executed for the software program during the period of time to be a sum of the calculated values for each of the identified protection domains (col. 6, lines 15-19 "Obtaining the final value of the loop control variable and calculating the number of iterations based on the initial and final values of the loop counter variable and the step size"); and if all of the

instruction holes that occurred during the period of time were attributable to the software program, calculating a value for the number of instruction holes for the processor that are attributable to the software program during the period of time to be the determined change in the value of the instruction holes counter (col. 6, lines 15-19 "Obtaining the final value of the loop control variable and calculating the number of iterations based on the initial and final values of the loop counter variable and the step size"), calculating a value for the number of instruction holes that are attributable to the software program during the period of time by all of the identified processors to be a sum of the calculated values for each of the identified processors (col. 6, lines 15-19 "Obtaining the final value of the loop control variable and calculating the number of iterations based on the initial and final values of the loop counter variable and the step size"); and determining the number of instruction slots available for execution of the instructions of software program during the period of time to be a sum of the determined number of instructions executed for the software program during the period of time and of the calculated value for the number of instruction holes that are attributable to the software program during the period of time (col. 6, lines 15-19 "Obtaining the final value of the loop control variable and calculating the number of iterations based on the initial and final values of the loop counter variable and the step size"); and presenting to a user an indication of the determined number of executed instructions for each of the periods of time and an indication of the determined number of available instruction slots for each of the periods of time (col. 2, lines 20-24 "graphically displaying execution profile information including the maximum value, the average value, the minimum value, and the standard deviation of the execution time of each routine").

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the method of calculating a value for the number of instructions executed for the software program during the period of time, and determining the number of instructions executed for the software program during the period of time to be a sum of the calculated values for each of the identified protection domains, and presenting to a user an indication of the determined number of executed instructions for each of the periods of time and an indication of the determined number of available instruction slots for each of the periods of time as taught by Asai in corresponding to method of analyzing trace information as taught by the combination system by Alexander and admitted prior art. The modification would be obvious because of one of ordinary skill in the art would be motivated to calculate a value for the number of instructions executed and indicating user that number of instructions has been executed and number of instructions are determined to be executed to provide an improved method of visualizing the results of performance monitoring and analysis in a parallel computing system as suggested by Asai (col. 1, lines 52-56).

Neither Alexander nor admitted prior art nor Asai explicitly disclose determining from the trace information a number of instruction slots available for execution of the instructions of software program during the period of time.

However, Ishikawa discloses in an analogous computer system determining from the trace information a number of instruction slots available for execution of the instructions of software program during the period of time (col. 2, lines 51-56 "a delayed typed instruction… executed at a time later than issuance of the branch instruction… one machine cycle.. an

instruction slot... the branch instruction is filled... valid instruction by means of an instruction... performed by a compiler").

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the method of determining available instructions slots for execution as taught by Ishikawa into the method of analyzing trace information as taught in the combination system by Alexander, admitted prior art, and Asai. The modification would be obvious because of one of ordinary skill in the art would be motivated to determining available instructions slots for execution to minimize the decrease of performance of a system as suggested by Ishikawa (col. 2, lines 48-49).

Claims 56, 57, 58, 78, 89, 102, and 129 are the system/device claim corresponding to method claim 1 and rejected under the same rational set forth in connection with the rejection of claim 1 above.

Claims 59, 60, 61, 77, 88, 101, and 128 are the computer program product claim corresponding to method claim 1 and rejected under the same rational set forth in connection with the rejection of claim 1 above.

Per claims 3-5:

Neither Alexander nor Asai nor Ishikawa explicitly disclose for at least one of the identified protection domains for at least one of the periods of time, there are no variable values in the trace information indicating a value for the issued instructions counter of that protection domain at an

end of that period of time (Applicant's specification, prior art; page 2, lines 1-2 "Each processor… have processor-specific counters reflecting the current processor state").

However, admitted prior art discloses in an analogous computer system wherein, for at least one of the identified protection domains for at least one of the periods of time, there are no variable values in the trace information indicating a value for the issued instructions counter of that protection domain at an end of that period of time (Applicant's specification, prior art; page 2, lines 1-2 "Each processor… have processor-specific counters reflecting the current processor state").

The feature of having the counter to track the instructions would be obvious for the reasons set forth in the rejection of claim 1.

Neither Alexander nor admitted prior art nor Ishikawa explicitly disclose wherein the determining of a second value for the issued instructions counter of that protection domain at the end of that period of time includes estimating the second value based on an extrapolation between earlier and later values for that issued instructions counter.

However, Asai discloses in an analogous computer system wherein the determining of a second value for the issued instructions counter of that protection domain at the end of that period of time includes estimating the second value based on an extrapolation between earlier and later values for that issued instructions counter (col. 6, lines 15-19 "Obtaining the final value of the loop control variable and calculating the number of iterations based on the initial and final values of the loop counter variable and the step size").

The feature of determining the final value for the protection domain with a counter to track the issued instructions would be obvious for the reasons set forth in the rejection of claim 1.

Per claims 13-16, 110-113, and 123-126:

The rejection of claims 1 and 103 are incorporated, respectively, and further, neither Alexander nor admitted prior art nor Ishikawa explicitly disclose presenting an indication of a logical code block of the software program that was executing during that period of time.

However, Asai discloses in an analogous computer system presenting an indication of a logical code block of the software program that was executing during that period of time (col. 4, lines 18-25 "A source code position analyzer 19 searches the source code to find the code position (or line number) corresponding to the item designated by the user... a source code browser 20 extracts the relevant part of the source code 15 and shows the extracted source code in a source code window 23 on the screen of the display unit 21").

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the method of displaying the an indication of the source code using an position analyzer as taught by Asai into the method of analyzing trace information as taught in the combination system by Alexander, admitted prior art, and Ishikawa. The modification would be obvious because of one of ordinary skill in the art would be motivated to display the indication of a code that is being executed to monitor the code being executed logically as suggested by Asai (col. 1, lines 52-56).

Per claims 17, 82, and 114:

The rejection of claims 1, 79, and 103 are incorporated, respectively, and further, Alexander disclose:

- presenting at least some of the variable values from the trace information in a tabular format (col. 5, line 43 "data... presented to a user in tabular form... shown in FIG. 5").

Per claims 22, 23, 30, 118, 119, and 126:

The rejection of claim 1 and 103 are incorporated, respectively, and further, Alexander disclose:

- wherein information to be presented is determined based on a specification provided by a user (col. 3, lines 41-42 "user interface adapter" and col. 5, lines 41-42 "tree structure... data... pictorially presented to a user").

Per claims 24 and 120:

The rejection of claim 1 and 103 are incorporated, respectively, and further, Alexander disclose:

- presenting information about memory references performed during at least one of the periods of time (col. 7, lines 2-4 "trace data... stored for... applications indefinitely... bounded memory requirement").

Per claims 25 and 121:

Neither Alexander nor Asai nor Ishikawa explicitly disclose presenting information about FLOPS performed during at least one of the periods of time.

However, admitted prior art discloses in an analogous computer system presenting information about FLOPS performed during at least one of the periods of time (page 10, lines 16-17 "related to the execution (e.g., memory latency, total execution time, or the number and rate of executed FLOPS, memory references, or invocations of a particular function)").

The feature of having the counter to track the instructions would be obvious for the reasons set forth in the rejection of claim 1.

Per claims 26-29, 122, 123, 124, and 125:

The rejection of claim 1 and 103 are incorporated, respectively, and further, neither Alexander nor Asai nor Ishikawa explicitly disclose presenting information about a number of the threads in existence during at least one of the periods of time.

However, admitted prior art discloses in an analogous computer system presenting information about a number of the threads in existence during at least one of the periods of time (Applicant's specification, prior art; page 10, lines 21-24 "it may be of interest to have information related to the threads for the task, such as the number of task threads executing the number of task threads blocked, the number of task threads ready and waiting to be executed, and the number of threads contending for a lock").

The feature of having the counter to track the instructions would be obvious for the reasons set forth in the rejection of claim 1.

Per claim 75 and 76:

The rejection of claim 62 is incorporated, and further, Alexander disclose:

- wherein the first computer provides multiple sources of information related to hardware aspects of the first computer that vary with execution of programs on the first computer, and wherein the distinct types of trace information for some of the types of events includes at least one of the sources of information related to the hardware aspects (col. 1, lines 28-31 "Performance tools may be implemented in hardware or software. Hardware performance tools are usually built into the system. Software performance tools may be built into the system or added at a later point in time")

6. Claim **33-38, 43-46, 63-72, 80-82, 85, 91, 93,** rejected under 35 U.S.C. 103(a) as being unpatentable over Alexander, admitted prior art, Asai, Ishikawa in view of US Patent No. 5,774,728 to Breslau et al., hereinafter called Breslau.

Per claims 33 and 63:

The rejection of claim 1 and 62 is incorporated, respectively, and further, neither Alexander nor admitted prior art nor Asai nor Ishikawa disclose wherein the software program is an executable version of a source code program such that execution of the executable version will generate the indicated trace information.

However, Breslau discloses in an analogous computer system the software program is an executable version of a source code program such that execution of the executable version will generate the indicated trace information (col. 7, lines 43-50 "The <u>source code</u> is parsed… to one

execution environment. During each iteration, a different target version is created (e.g., target versions 25, 27 & 29 of FIG. 2)").

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate method of generating the different target versions of the program as taught by Breslau into the method of analysis of trace information at the time of execution in a combination system as taught by Alexander, admitted prior art, Asai and Ishikawa. The modification would be obvious because of one of ordinary skill in the art would be motivated to generate the different executable versions for the target the devices to work on different execution environment as suggested by Breslau (col. 10, lines 54-63).

Per claims 34-35, 43, 44, 46, 64, 65, 66, 67, 80, 81, 82, 85, 91, and 93:

The rejection of claims 33, 1, 62,79, and 90 are incorporated, respectively, and further, neither Alexander nor admitted prior art nor Asai nor Ishikawa disclose generating the executable version by compiling the source code program such that a group of instructions is added to the source code program at a location specified by the user, the added instructions when executed to generate trace information for a type of event specified by the user.

However, Breslau discloses in an analogous computer system generating the executable version by compiling the source code program such that a group of instructions is added to the source code program at a location specified by the user, the added instructions when executed to generate trace information for a type of event specified by the user (5774728; col. 2, lines 13-19 "each target version... compiled with a selectable target compiler... informed of the execution environment of the target version... selectable target compiler of the execution environment may

be performed by placing a compiler directive in each target version that specifies the execution environment of the target version").

The feature of generating the executable versions would be obvious for the reasons set forth in the rejection of claim 33.

Per claims 36-38, 68, 69, 70, 71, and 72:

The rejection of claim 33, 66, and 69 are incorporated, respectively, and further, neither Alexander nor admitted prior art nor Asai nor Ishikawa disclose wherein the automatically identified locations include beginnings of functions having more lines than a threshold.

However, Breslau discloses in an analogous computer system wherein the automatically identified locations include beginnings of functions having more lines than a threshold (5774728; col. 1, lines 66-67 to col. 2, line 1 "automatically compiling the first section and the second section for their corresponding execution environments using their section identifiers").

The feature of automatically identified locations would be obvious for the reasons set forth in the rejection of claim 33.

7. Claims 39 and 40 are rejected under 35 U.S.C. 103(a) as being unpatentable over Alexander, admitted prior art, Asai, Ishikawa in view of US Patent No. 5,812,811 to Dubey et al., hereinafter called Dubey.

Per claim 39,40:

The rejection of claim 1 is incorporated, and further, neither Alexander nor admitted prior art nor Asai nor Ishikawa disclose wherein the instructions related to parallelizing the execution of the executable version are instructions related to a fork.

However, Dubey discloses in an analogous computer system the instructions related to parallelizing the execution of the executable version are instructions related to a fork (col. 4, lines 54-57 "invention discloses… Fork-Suspend instructions… added to the instruction set of the CPU, and are inserted in a program prior to run-time to delineate potential future threads for parallel execution").

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the method of having the instructions related to the fork as taught by Dubey into the method of analysis of trace information at the time of execution in a combination system as taught by Alexander, admitted prior art, Asai and Ishikawa. The modification would be obvious because of one of ordinary skill in the art would be motivated to have instructions related to fork to enhance a single thread to simultaneously fetch, decode, executed in multiple program locations as suggested by Dubey (col. 4, lines 32-48).

8. Claims 7, 47, 48, 86, and 87 are rejected under 35 U.S.C. 103(a) as being unpatentable over Alexander, admitted prior art, Asai, Ishikawa in view of US Patent No. 5,594,904 to Linnermark et al., hereinafter called Linnermark.

Per claims 7, 47, 48, 86, and 87:

The rejection of claim 1, 46, 85 are incorporated, respectively, and further, neither Alexander nor admitted prior art nor Asai nor Ishikawa disclose wherein the other executing program is a background daemon.

However, Linnermark discloses in an analogous computer system the other executing program is a background daemon (col. 5, lines 51-54 "The detection of events is made possible by the insertion of code sequences, i.e., daemons 46, at any level in the software as shown by the small circles distributed through the application 40, operating system 43, and the kernel 45").

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the method of background daemon executing the program as taught by Linnermark into the method of analysis of trace information at the time of execution in a combination system as taught by Alexander, admitted prior art, Asai and Ishikawa. The modification would be obvious because of one of ordinary skill in the art would be motivated to use the background daemon as an executing program to detect small instructions executed on a processor as suggested by Linnermark (col. 3, lines 3-18).

Allowable Subject Matter

9. Claims 2, 41, 42, 45, 55, 72, 73, 84, 100, 49-54, and 96-99 are objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

The cited prior art taken alone or in combination fail to teach the method for analyzing performance of execution of a task in a computer system including multiple threads of the task, the combination of analyzing generated trace information *by determining from the trace information at least one swap event that occurred in the protection domain during the period of time such that the software program is swapped into the protection domain so as to commence execution of the software program, retrieving for each of the determined swap events an associated value in the trace information of the issued instructions counter of the protection domain, using the retrieved associated values to calculate the value for the number of instructions executed for the software program during the period of time by the protection domain to include only increments to the issued instructions counter that occurred while the software program is swapped into the protection domain* as recited in the dependent claim 2.

The cited prior art taken alone or in combination fail to teach the method for analyzing performance of execution of a task in a computer system including multiple threads of the task, the combination of analyzing generated trace information by *creating a descriptor object for the specific group of instructions that are added to the source code of the program at the time of execution* as recited in the dependent claims 41, 42, 45, 55, 72, 73, 84, and 100.

The cited prior art taken alone or in combination fail to teach the method for analyzing performance of execution of a task in a computer system including multiple threads of the task, the combination of analyzing generated trace information by *extracting execution information corresponding to the events that occurred during the execution, using a description of specified types of events to identify groups of trace information each reflecting execution of a group of added instructions, using the identified groups and created mapping to extract current values for execution information and modifying current values of events so as to be normalized with respect to beginning of the execution* as recited in the dependent claims 49-54, and 96-99.

Conclusion

10. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

The following patent is cited to further show the state of the art with respect to analysis of trace information.

US Patent No. 5,548,717 to Wooldridge et al.

US Patent No. 6,145,121 to Levy et al.

US Patent No. 6,282,701 to Wygodny et al.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Satish S. Rampuria whose telephone number is 703-305-8891. The examiner can normally be reached on 8:30 am to 5:00 pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (703) 305-9662. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Satish S. Rampuria SR.

Patent Examiner

Art Unit 2124

06/14/2004

ANIL KHATRI
PRIMARY EXAMINER